

OSG Services for Campuses

Brian Bockelman

PATh Fabric of Capacity Services/Production Services

**Investigator,
Morgridge Institute for Research**



This project is supported by National Science Foundation under Cooperative Agreement OAC-2030508. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.



- Today I want to outline three services relevant to a campus community:
 - Sharing Compute Resources (the “Compute Entrypoint”).
 - Providing an Access Point to the OS Pool and OS Data Federation (origin).
 - Integrating data resources as caches in the data federation.
- Let’s start with resource sharing!



OSG is a consortium dedicated to the advancement of all of open science via the practice of distributed High Throughput Computing (dHTC), and the advancement of its state of the art.

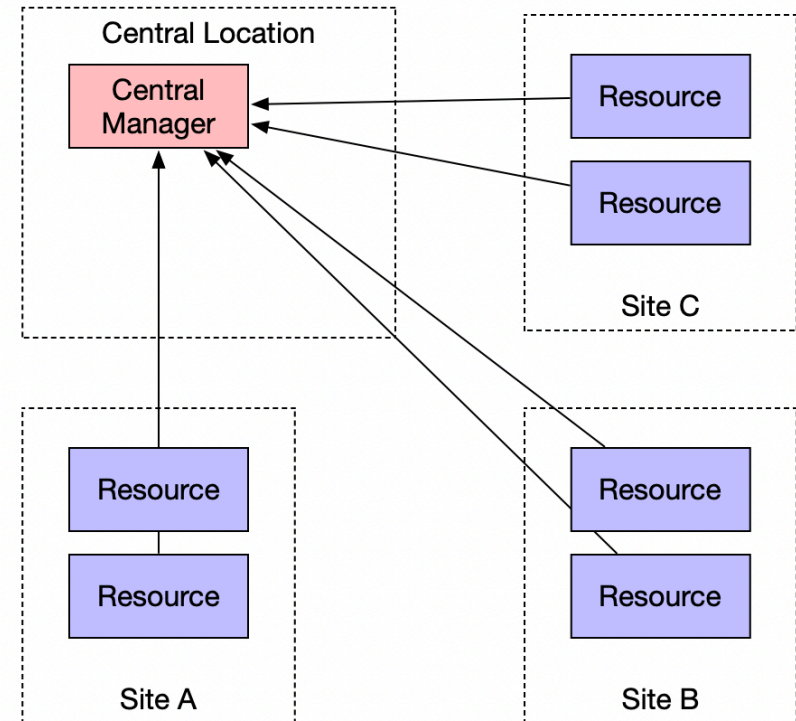
dHTC enables us to effectively enable resource sharing across more than a hundred sites in the US!



Resource Sharing



- OSG shares resources via the concept of an “overlay pool”.
 - Disparate worker node resources are allocated (think: starting a VM).
 - Some piece of software (“pilot”) starts on the worker node which subsequently connects to a central pool.
 - Work – batch jobs! - from the outside is pulled down to the worker node and executed. These are “payload” jobs.
- Technologies we’ll see today:
 - [HTCondor](#) manages the jobs and central pool.
 - [glideinWMS](#) helps decide where and when to allocate resources.
 - Batch systems: that’s up to you!



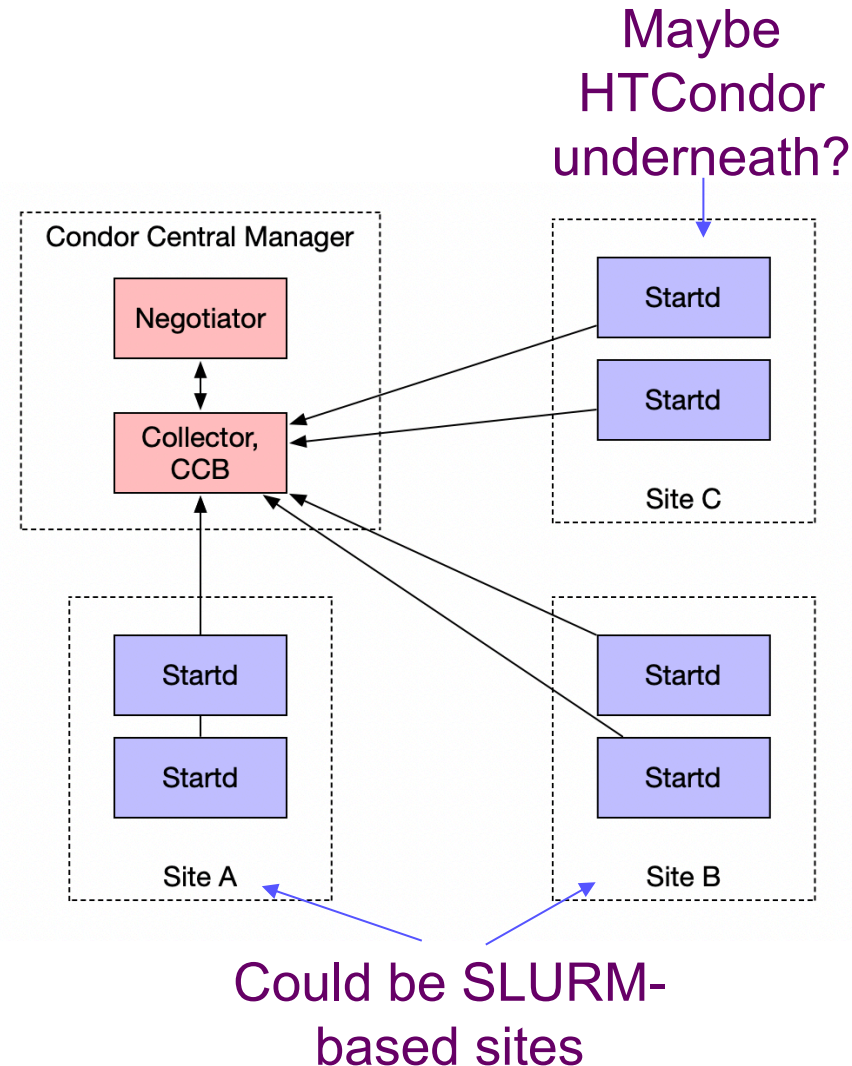


Now with Jargon!



In HTCondor Jargon:

- The software the pilot deploys on the site's worker node is called the **startd**.
- The central manager has three primary components:
 - **Collector**: daemon where all the resource descriptions are uploaded from the worker node.
 - **Condor Connection Broker (CCB)**: Used to manage network connections, allowing **startd** to be behind a NAT.
 - **Negotiator**: Implements policy and allocates the share of resources to different users.





Important Observations



- OSG heavily leverages HTCSS (HTCondor) for constructing its resource sharing.
 - Your *campus* does not need to run HTCSS.
 - Of course, ‘HTCondor on top of HTCondor’ works quite well
 - OSG supports integration with SLURM, SGE, and PBS as a batch system.
 - Non-batch system integrations have included Kubernetes (quite successful) and OpenStack (less popular).
- An important OSG principle is site autonomy. Your campus makes the decisions, not OSG.
 - Obviously, decisions have impact. For example, almost all GPU jobs run inside containers. If your campus doesn’t support containers, OSG may not have jobs to run.



Processes on the worker



```
condor_startd -f
└─condor_starter -f -a slot1 test-006.t2.ucsd.edu
    └─condor_exec.exe /var/lib/condor/execute/dir_146125/condor_exec.exe -v std -name gfactory_instance -entry OSG_US_UCSD_TEST_CE_004 -clientname test.test -schedd...
        └─condor_startup. /var/lib/condor/execute/dir_146125/glide_q5Xf8G/main/condor_startup.sh glidein_config
            └─condor_master -f -pidfile /var/lib/condor/execute/dir_146125/glide_q5Xf8G/condor_master2.pid
                └─condor_procd -A /var/lib/condor/execute/dir_146125/glide_q5Xf8G/log/procd_address -L /var/lib/condor/execute/dir_146125/glide_q5Xf8G/log/ProcLog -R ...
                    └─condor_startd -f
```

- We overlay a batch system inside a batch system. This is not a common thing to do!
- Beyond the batch system processes, the running pilot has a few components:
 - A shell wrapper to startup the pilot,
 - Condor helper daemons,
 - The condor worker node processes and the process representing the running job
 - And finally, any processes that are part of the payload job!
- **Warning: more processes than you typically see on the worker node.**

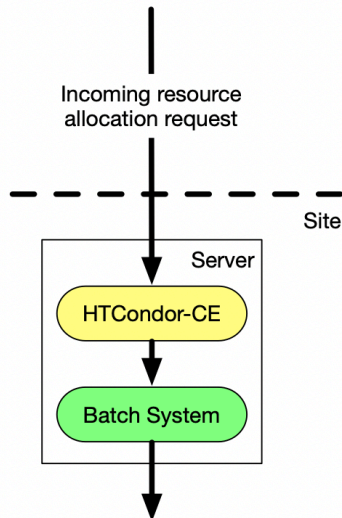


Resource Requests

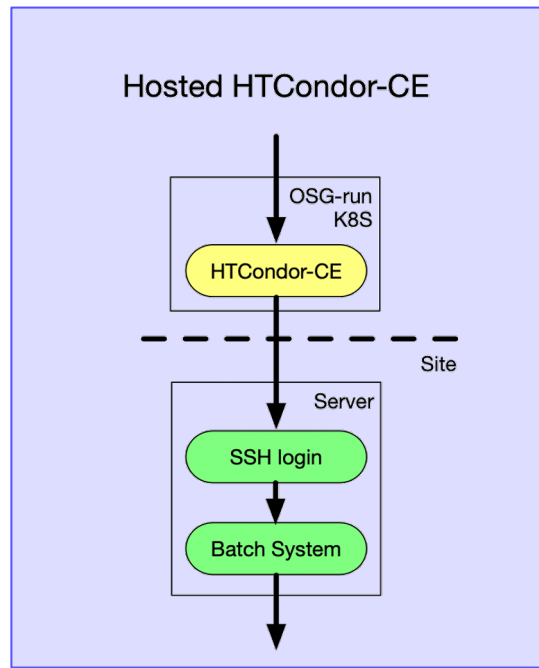


- To start a pilot at the site, some service is needed to connect the site's batch system to the external world.
 - This is a “Compute Entrypoint”, or “CE”.
 - We use the “HTCondor-CE” implementation.
 - Three supported models: traditional onsite deploy (by site), Hosted CE (run by OSG), or an onsite hosted CE (run by OSG).

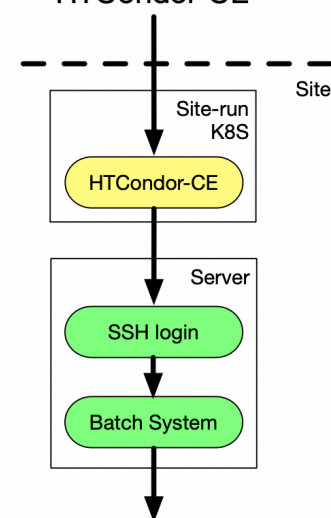
Traditional HTCondor-CE



Hosted HTCondor-CE



Onsite Hosted HTCondor-CE



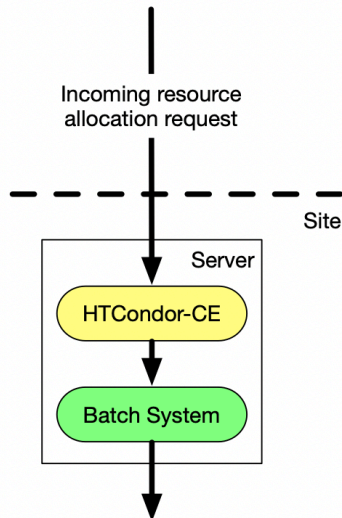


Resource Requests

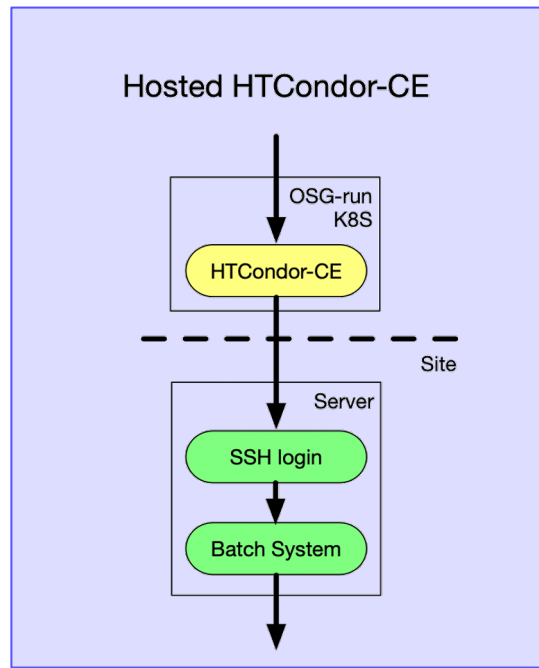


- Today we'll cover the Hosted CE. The different deployment types:
 - Determine how much **configuration the site controls** (traditional deploy offers most control).
 - Determines the **total site admin time invested** (hosted option is the minimal option).

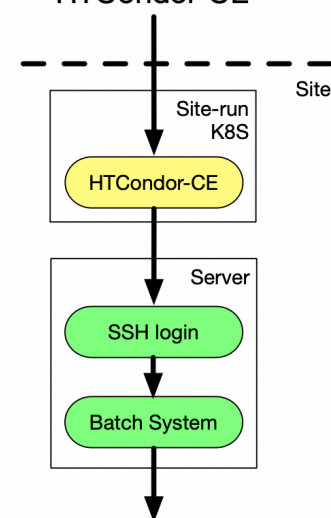
Traditional HTCondor-CE



Hosted HTCondor-CE

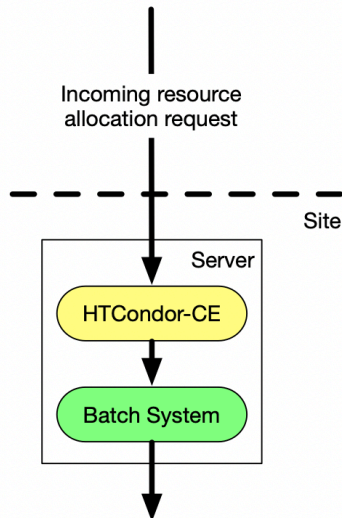


Onsite Hosted HTCondor-CE

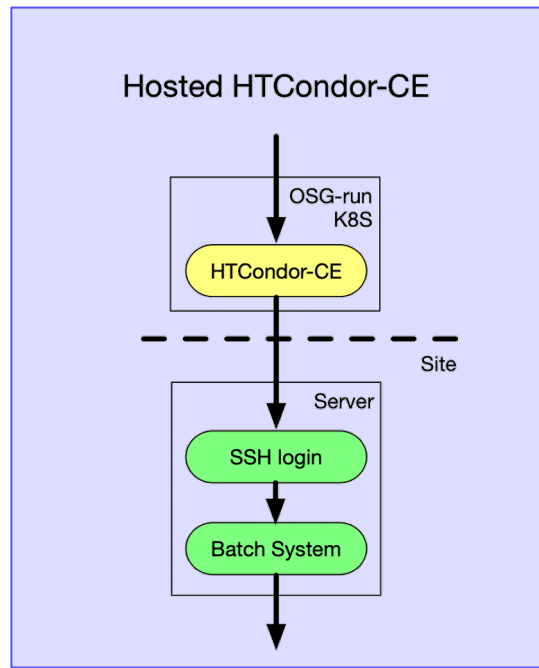


- Hosted CE: Here, 'hosted' means 'run by OSG'.
 - There's no charge for this service – supported by the NSF-funded PATH project.
 - Works for sites large and small.

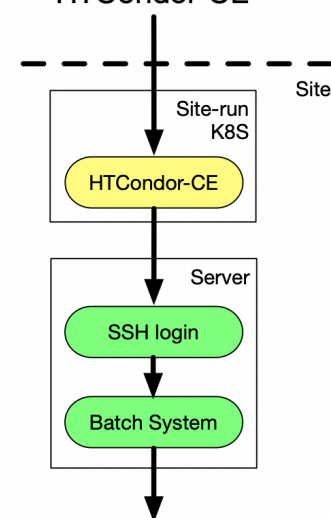
Traditional HTCondor-CE



Hosted HTCondor-CE



Onsite Hosted HTCondor-CE



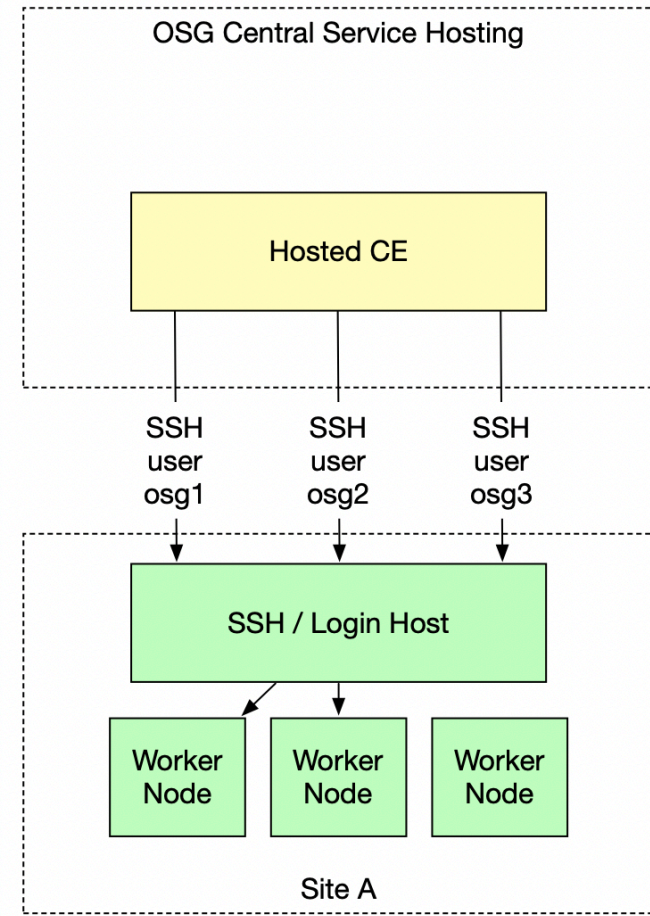


Allocating Resources



“Resources Allocations” are really batch jobs.

- These batch jobs arrive from the hosted CE to the site login host via SSH.
 - One CE per SSH host.
 - **Each supported community is a separate Unix account.**
- You can ban or prioritize individual communities without involving OSG as each community is a distinct batch user.

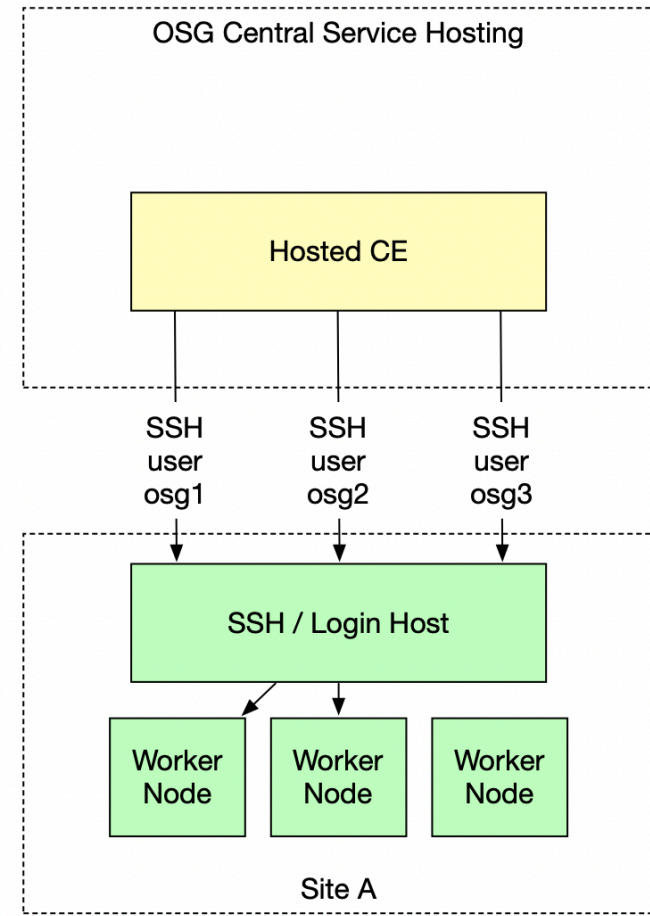




Allocating Resources



- The OS Pool is a specific community operated by OSG.
 - This includes XSEDE allocations, scientific collaborations -- all under the umbrella of 'open science' and research.
 - We have simple demographics and accounting for this community available to you.
- OSG is here to facilitate resource sharing, not demand specific resource allocations (**autonomy!**).
 - You can decide whether you share resources with the OS Pool, LIGO, CHTC (UW-Madison), IceCube, LHC, or others.
- We can help with batch system configuration to implement your desired policy!

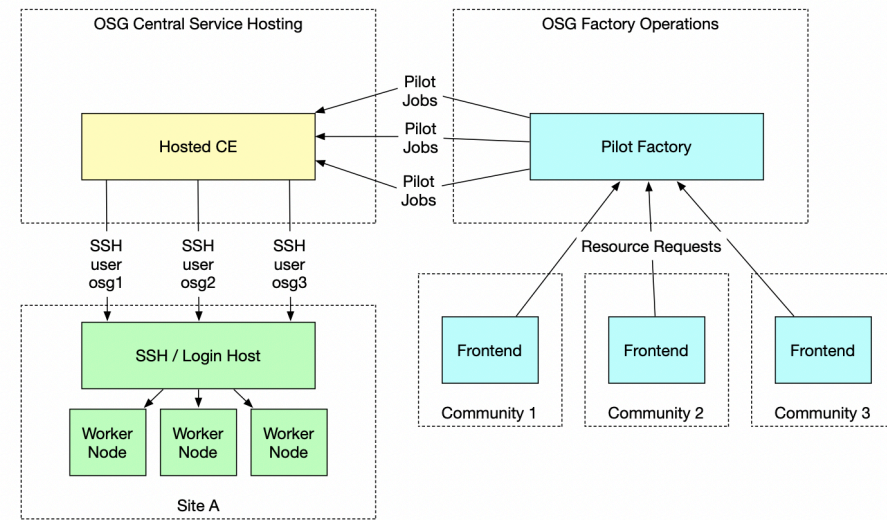


Allocating Resources



The OSG “factory” creates pilot jobs for each CE to submit to the local batch system.

- Most, but not all, communities use the OSG-run factory. Some run their own.
- Each community has a “frontend,” which determines the resource requests per site based on their current job load and profile.
- The factory translates those resource requests to pilot jobs to various CEs.
 - The hosted OSG-CE will transform these to jobs appropriate for your site’s batch system and submit over SSH.





Example – Processes on the SSH host



```

root      40988  0.0  0.0 137360  4324 ?        Ss   11:01   0:00 \_ sshd: ligouser [priv]
ligouser  40992  0.0  0.0 137360  2036 ?        S    11:01   0:00 \_ \_ sshd: ligouser@notty
ligouser  40993  0.0  0.0 144956  3180 ?        Ss   11:01   0:00 \_ \_ /home/ligouser/bosco/glite/bin/batch_gahp
root      41149  0.0  0.0 137360  4324 ?        Ss   11:01   0:00 \_ sshd: ligouser [priv]
ligouser  41153  0.0  0.0 137828  2536 ?        S    11:02   0:00 \_ \_ sshd: ligouser@notty
ligouser  41158  0.0  0.0 106124  1440 ?        Ss   11:02   0:00 \_ \_ /bin/bash -l -c echo Allocated port 46057 for remote
forward to l>&2 ; CONDOR_CONFIG=~ /bosco/glite/etc/condor_config.ft-gahp ~ /bosco/glite/bin/condor_ft-gahp -f
ligouser  41302  0.0  0.0  52356  4788 ?        S    11:02   0:00 \_ \_ /home/ligouser/bosco/glite/bin/condor_ft-gahp -f

```

- On the SSH login host, the site will see two persistent SSH connections:
 - One for the processes that interacts with the batch system (`batch_gahp`).
 - One for file movement (`condor_ft-gahp`).
- These will occasionally launch other processes (e.g. `qsub`).



Sharing Compute Resources - Other Activities



- What else occurs at the site?
 - You'll see one set of processes on the submit host for each user account setup for OSG.
 - Worker nodes need outgoing network connectivity to the central pool and submit hosts. NAT is fine!
 - The outgoing IP addresses will vary from community to community.
 - Only inbound connections go to SSH host.
 - For non-HTCondor sites, a shared filesystem is needed to move startup scripts and logs.
- Individual science communities (such as LHC) may require additional services; let us know who you want to support, and we can provide more details.

No Batch System? No Problem!



- Added in late 2020 was the ability for campuses to *backfill* resources with OSG.
 - Campuses can launch a container directly on any available resource.
 - An especially popular option for Kubernetes-based clusters.
 - This is useful for **backfill** mode.
 - As OSG didn't submit the resource request, no guarantee there's available jobs to run.
 - That's fine if the resource is otherwise idle.
- See documentation for more details.

- To get started with the hosted CE:
 - Contact support@opensciencegrid.org to schedule a discussion with the OSG team.
 - This helps us set our goals for integrating the site and determine what services we should target.
 - For hosted CEs, we'll ask you to fill in the [cluster integration questionnaire](#).
 - This simply gets us some basic technical facts.
- Feel free to contact us early in the design process – you don't need to have a batch system or hardware to start planning!



SECURITY ON OSG

- Security is an important process required to establish mutual trust between parties!
 - OSG acts as a trusted middleman.
 - OSG establishes identity and relationships with both sites and communities.
- Sites must know their computing resources are used for the advancement of science and engineering.
 - For example, The facilitation team meets with each OSG User over video to help establish identity.
- Communities must know their computing is performed on valid resources.

- A few examples of technical mechanisms in place to protect your site:
 - TrustedCI, NSF's Cybersecurity Center of Excellence, has performed an assessment of the security of the HTCondor-CE itself.
 - All of our containers are periodically rebuilt to pick up the latest RedHat security patches.
 - The SSH private key is accessible only to the OSG Operations team and will only be used from specific, narrow IP ranges.
 - OSG components use mutual X.509-based authentication to establish identities; all communication is encrypted.
 - All OSG services are periodically scanned for vulnerabilities.

Security – Everyone's Responsibility



- Sites, too, have responsibilities:
 - As necessary, participate in and be responsive during a security incident response.
 - Notify OSG in the case of a site-level incident affecting grid components.
 - Maintain a secure local environment; perform best-practice techniques such as:
 - Keep hosts patched and up-to-date.
 - Maintain traceability – keep logs of activity at the site.
 - Monitor network activity for suspicious traffic.
- Remember, OSG users are trusting the site with their data, credentials, and scientific work!



ACCESS POINTS



OS Pool Access Points



- The OS Pool supports multiple access points.
 - Each access point can be independently administered.
- OSG operates the “OSG Connect” access points on the OS Pool.
- Why do people run their own APs?
 - Integration with local data sources. Jobs can read directly from the local filesystem.
 - Transparent access to local resources
 - Allow users to login with the computing center credentials.
 - In-house user support.



OS Pool Access Points

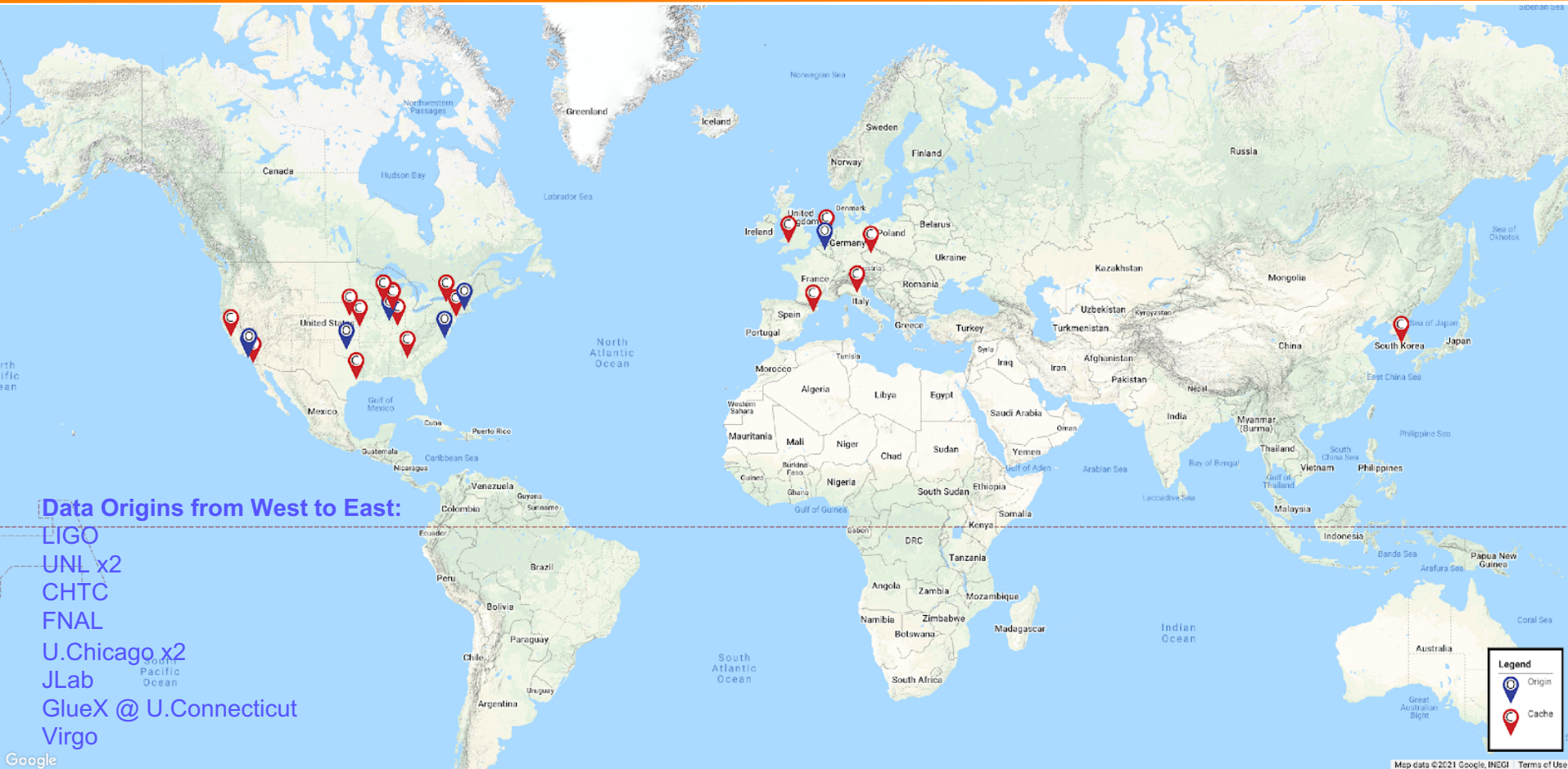


- What resources are needed to stand up an access point? For hardware:
 - Modest memory & CPU (minimum 16GB RAM, 4 cores).
 - Incoming network access (TCP port 9618 open).
 - Fast storage (SSD) for job database.
- Most often scale is driven by your users, not the minimum requirements. Production access points are often much larger.
- The access point is distributed as an RPM and a security token.
 - Beyond the obvious HTCondor scheduler service, a job accounting probe runs.
- The hardware resources are minimal – but operating a service is a significant human resource commitment.
 - Use support channels need to be setup. In-house expertise on HTCondor is necessary.
 - One way to build this expertise is to start with OSG Connect. You can “shadow” OSG facilitators when they work with users from your campus.



Open Science Grid

Data Access in OSG



17 Caches ... 6 of which are in R&E network backbone

10 Data Origins ... one of which is for all of open science



- The OS Data Federation “origin” service allows users to access campus data without needing an access point on campus.
- The software, based on XRootD connects a local POSIX file system to the data federation.
 - Clients connect directly to caches, not the origin. Caches take the brunt of the load, especially for repeated data accesses.
 - In some cases, the origins see hundreds of GB of access while the caches deliver hundreds of TBs.
 - The software is distributed as a Docker container or as RPMs.
- The origin provides coarse-grained authorization access; practical to set up ACLs for directories trees but not at a per-file level.



CONTAINERS & DATA

Sharing Resources More Effectively



- Integrating a Hosted CE is the simplest way to share resources.
 - Requires no site-run services and aims to minimize required site effort.
- By running additional services, your site will run more efficiently or be able to support a broader range of jobs
 - Example: effectively all our GPU jobs require container support at the site.
- Your site does more science – but more effort is involved!
 - Your local researchers may be the ones benefitting from these services.



Sharing Resources More Effectively

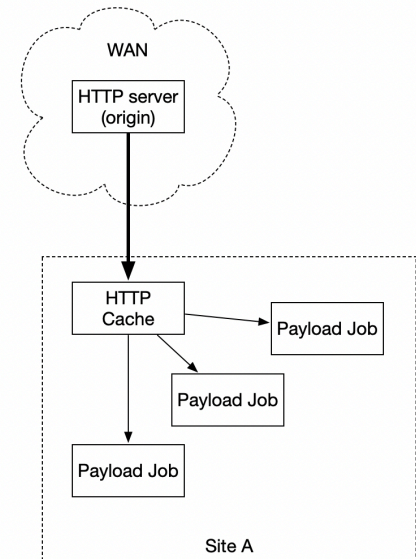


- Example campus-level services:
 - **HTTP Cache**: helps avoid frequent retransfer of many (small) resources over HTTP.
 - **Worker-node Container Distribution**: Global, read-only, caching filesystem for distributing containers and software (data is moved via the HTTP cache).
 - **Singularity**: Allows us to launch jobs inside containers. Relies on the container distribution mechanism.
- I'll include links to documentation; tackle these if desired (and after the basics are working).

Sharing Resources More Effectively - HTTP Cache



- A broad set of data – software, configurations, job inputs – can be moved to the worker node via HTTP. A significant amount is very frequently reused.
 - By placing a HTTP cache on-site, repeated data use only goes over the LAN instead of the WAN.
 - Any caching HTTP proxy can work for OSG. However:
 - Not all scale well in terms of concurrency.
 - Most are tuned for HTML files, not objects in the >1MB range.
 - We work with the Frontier project to support a special configuration of the venerable Squid software, frontier-squid. Monitoring, logging, and configuration are tuned specifically for OSG usage.
- [Sysadmin Documentation](#). Complexity level: **Easy**. Standalone service on dedicated host.
 - Can be run on local Kubernetes by OSG staff.

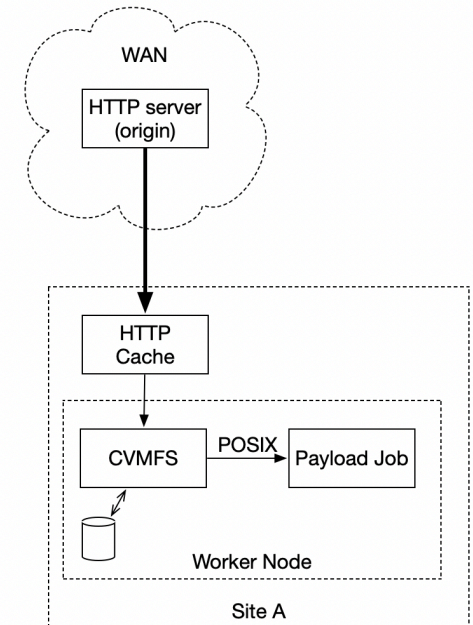




Sharing Resources More Effectively – Container Distribution (CVMFS)



- CVMFS is a global, integrity-checked, read-only POSIX filesystem. This achieves scale by distributing data through HTTP caches and a CDN.
 - Everything is cached – filesystem metadata and data – all the way to the local worker node. Data is moved to the worker node only on access.
 - In OSG, we use this as a mechanism to distribute science community containers and software. Given the popularity of containers, many jobs require this.
 - **Downside:** This is software that is run on the worker node, which adds complexity.
 - **Downside:** Implemented using FUSE and autofs, two tricky technologies.
 - **Good news on the horizon:** In newest CentOS 7, this can be done by the batch job completely unprivileged. Nothing to install or monitor on the worker node.
 - Expected in the next glideinWMS release.
- [Sysadmin Documentation](#). Complexity level: **Moderate**. Software that is installed on worker node & filesystem mounted.



Sharing Resources More Effectively – Containers (Singularity)



- Singularity is a container runtime that aims to fit the needs of running containers inside a batch system. No running daemon like Docker – just a process inside the batch job.
 - For full functionality, a `setuid` (extra privileges) binary is needed.
 - For OSG use cases – and on RHEL7 – we recommend using `unprivileged` (no `setuid`).
- The pilot will invoke Singularity prior to starting the payload; this way, the pilot sees the host operating system and the payload sees the container of its choice.
 - The containers are typically distributed via CVMFS.
- OSG provides targeted support for the “community edition” of Singularity.
- [Sysadmin Documentation](#). Complexity level: **Easy**. Single configuration file on worker node.



Sharing Resources More Effectively – Data Caches



- The Frontier-Squid software targets the distribution of “small-ish” objects – less than 1GB:
 - Is inefficient to use for files over 1GB.
 - Does not provide a mechanism to securely cache proprietary scientific data. (Note: OSG does not provide mechanisms suitable for HIPAA data.)
- We have a separate software (XCache, a special configuration of the XRootD software) to fill this role.
 - Designed for delivering 1-10GB of data to jobs where there is cache-friendly access and the total working set size of a workflow is <10TB.
 - Provides mechanisms to authenticate and authorize
 - In the end, still transfers data via HTTP / HTTPS.
- [Sysadmin Documentation](#). Complexity Level: **Moderate / Hard**.
 - Can be run on local Kubernetes by OSG staff.



Putting it all Together



- Some important take-homes for today:
 - Campuses can share resources in a batch system with OSG through a compute entriypoint.
 - OSG will host the CE so the site only needs a SSH login host, a batch system, and network.
 - OSG enables effective resource sharing using an overlay pool.
 - Campuses can run access points locally to provide OSG integration. Often this is done for data access or after successful use of OSG-Connect.
 - Other site level services – Squid, Singularity, CVMFS, or data caches – allow the campus to better share local resources.



Questions?

support@opensciencegrid.org

This material is based upon work supported by the National Science Foundation under Grant No. [2030508](#). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.



BACKUP SLIDES



RESOURCE ACCOUNTING



Resource Accounting for CC*



- CC* program aims to make available 20% of the resources to external communities.
 - Both “make available” and “20%” are surprisingly hard to define. How you interpret this is between you and NSF – not OSG’s business!
 - OSG helps provide input to this process by making some simple accounting numbers available.
- Unfortunately, accounting is surprisingly subtle on OSG:
 - **Pilot** accounting tells us what compute resources were made available via the batch system.
 - **Payload** accounting tells us how the communities use the allocated resources.
- Ideally, these are identical: **in practice, they are not!**
 - Example: a pilot may start up but find that all payload jobs are already gone.
 - In most cases, these numbers are within 10% of each other.



Other Accounting Gotchas

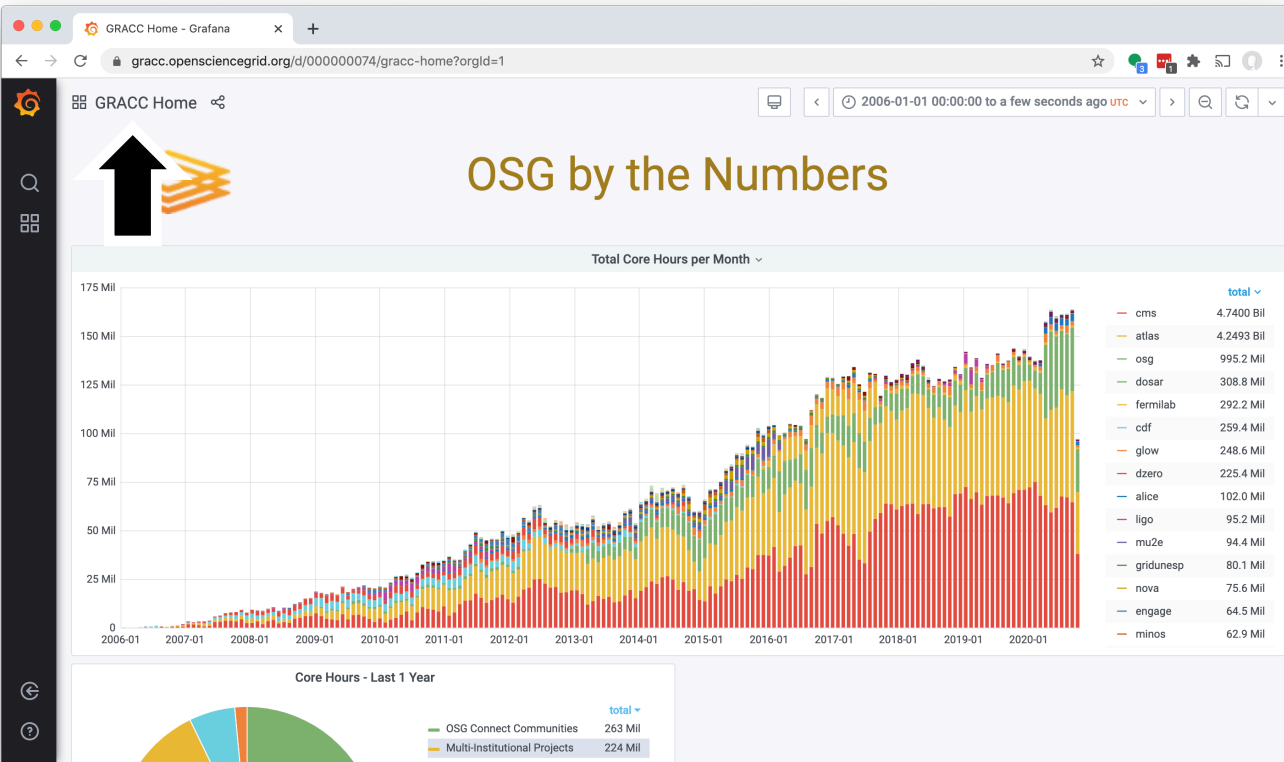


- A few other notable “gotchas”
 - Not all communities report the same details to OSG on how they use the resources.
 - **Example:** LHC community does not provide payload details.
 - Some communities utilize the OSG services to reach non-US sites.
 - **Example:** we only get payload information from European sites running IceCube, not pilot.
- Because this is complex, it’s useful to think carefully about what question you want to ask the system.
 - I’ll walk through a few screenshots on what I think is most important.

Grid Resource ACCounting (GRACC) portal



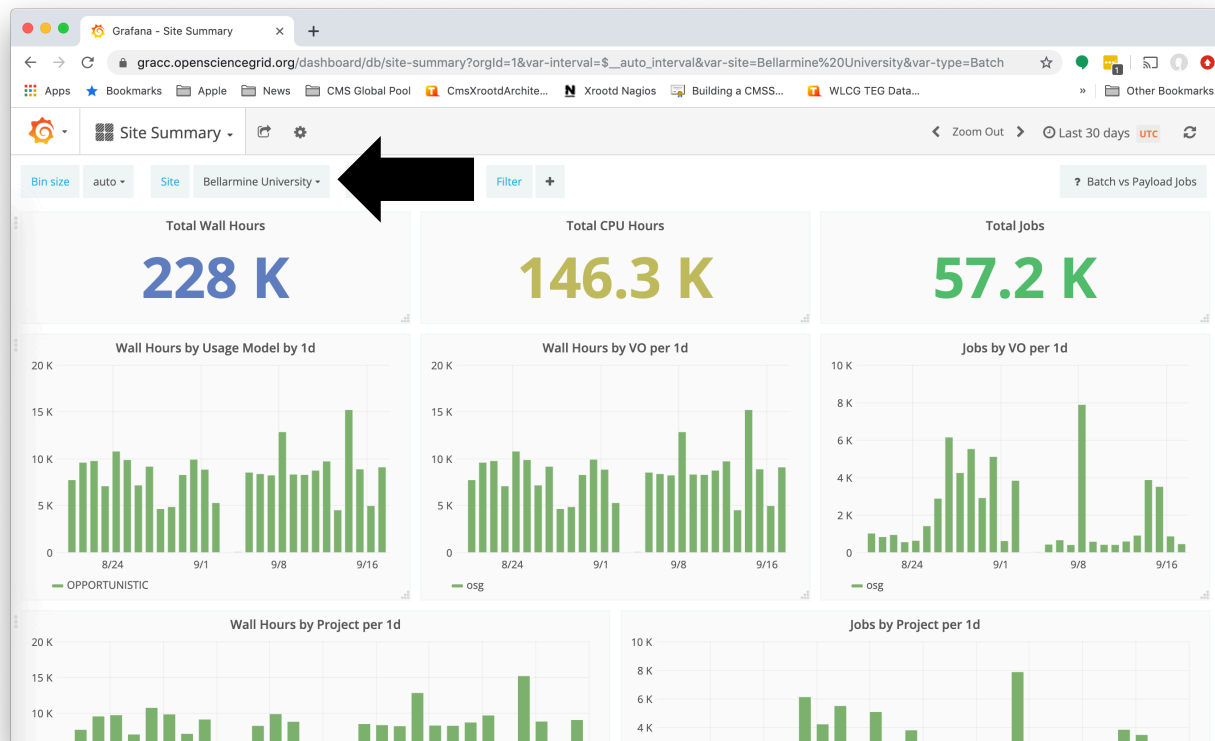
Accessible at
[https://gracc.
openscience
grid.org](https://gracc.opensciencegrid.org) or by
clicking on
“[explore our
accounting
portal](#)” on the
homepage.





Open Science Grid

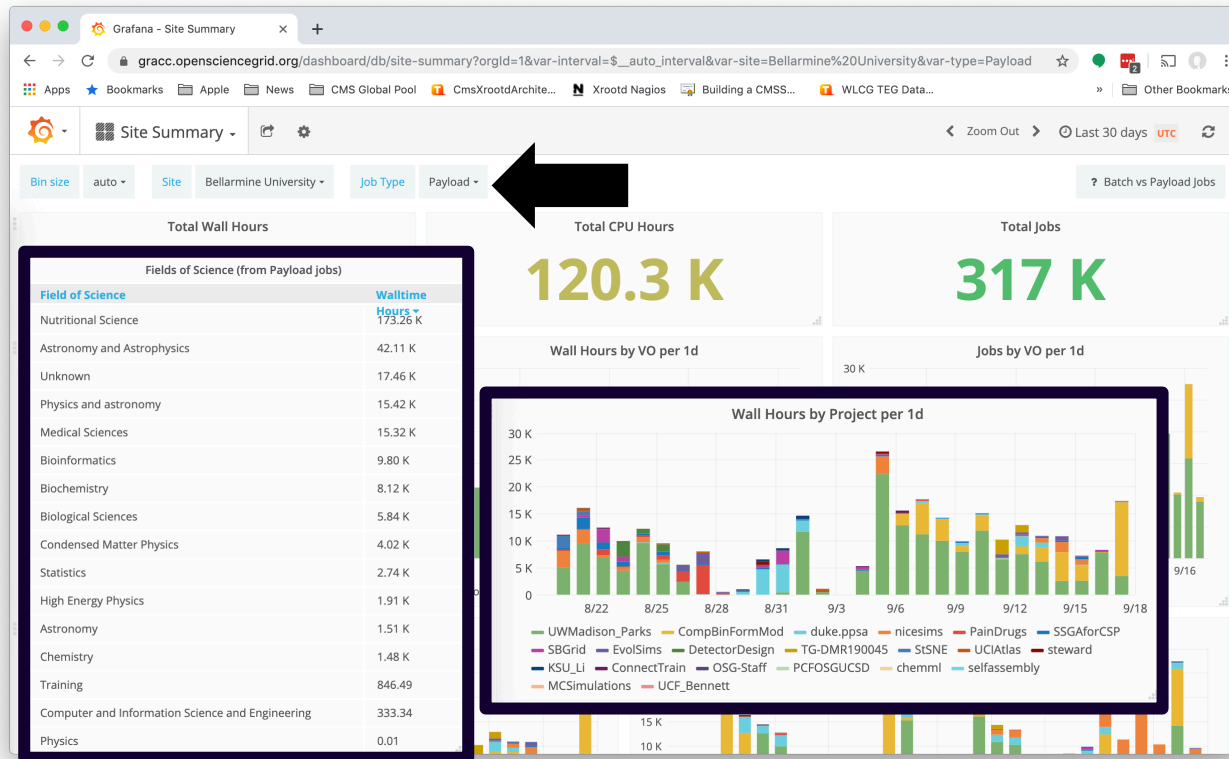
Grid Resource ACCounting (GRACC) portal



The “Site Summary” page defaults to all sites; use the drop-down to select your site name.

- You get to pick your site name. Names like “Bellarmine University” tend to be more descriptive than “KR-KISTI-GSDC-02”.
- This shows the view by community; in this case, only the special “osg” community was run.

Grid Resource ACCounting (GRACC) portal



- Switch to the “payload” job type to get information about the resource usage, including
- The projects names inside the community.
- The corresponding fields of science.